Transportation Air Quality and MOVES Training

Module 4: MYSQL Workbench and Queries

Note: This material is part of a five module training course prepared by the Texas A&M Transportation Institute (TTI) for the Texas Department of Transportation. Please review the training description document for further details and for TTI contact information

Objectives

- Overview of MYSQL Workbench/Query browser
- Exploring MOVES Databases with the MYSQL Query Browser
- Basic MySQL Commands
- Executing MySQL Scripts Using MOVES Interface
- Advanced MySQL Commands/Scripts
- Creating and Manipulating Data with MySQL

MYSQL Workbench Connection

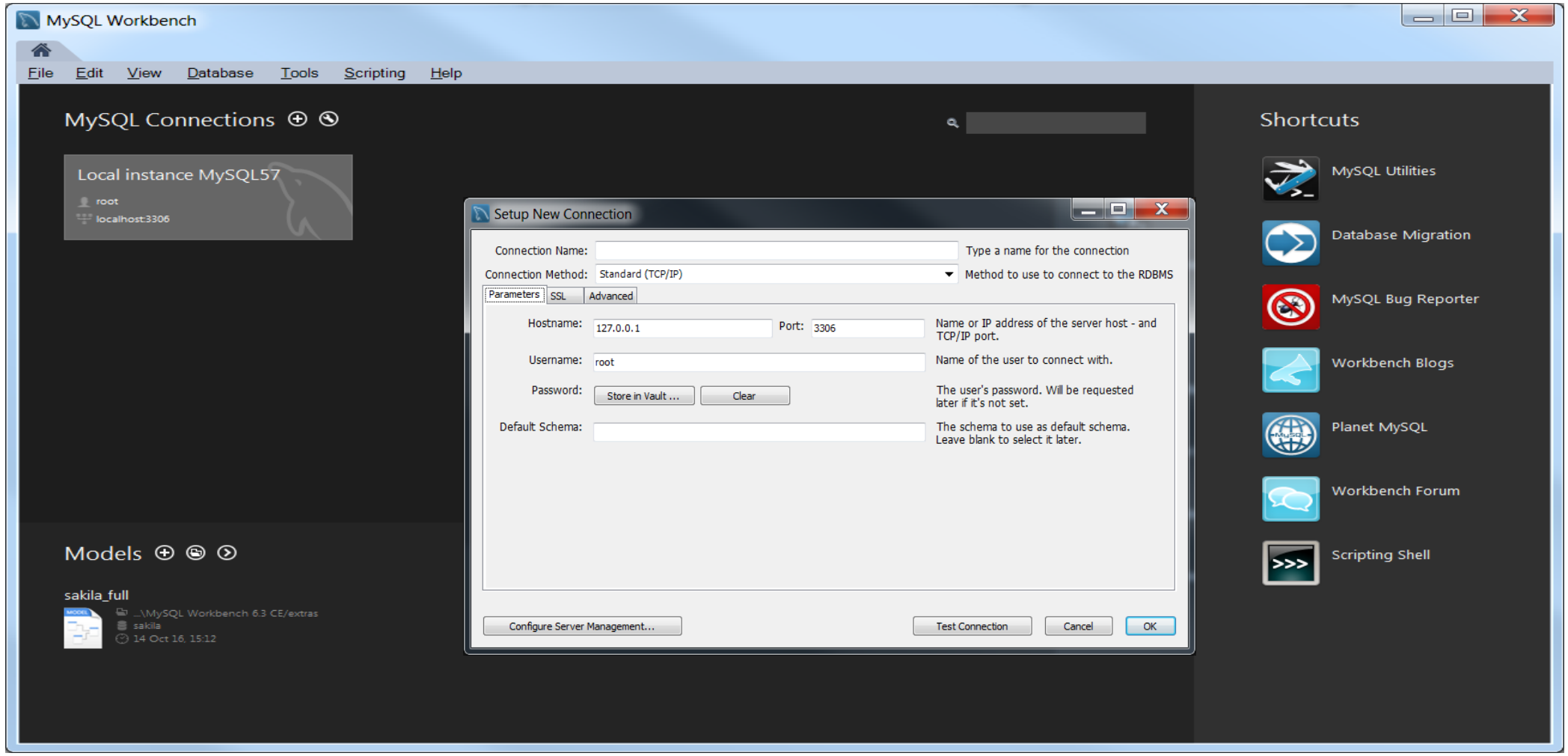
Start/Programs/MySQL/MySQL Workbench

Click *Local instance MySQL*

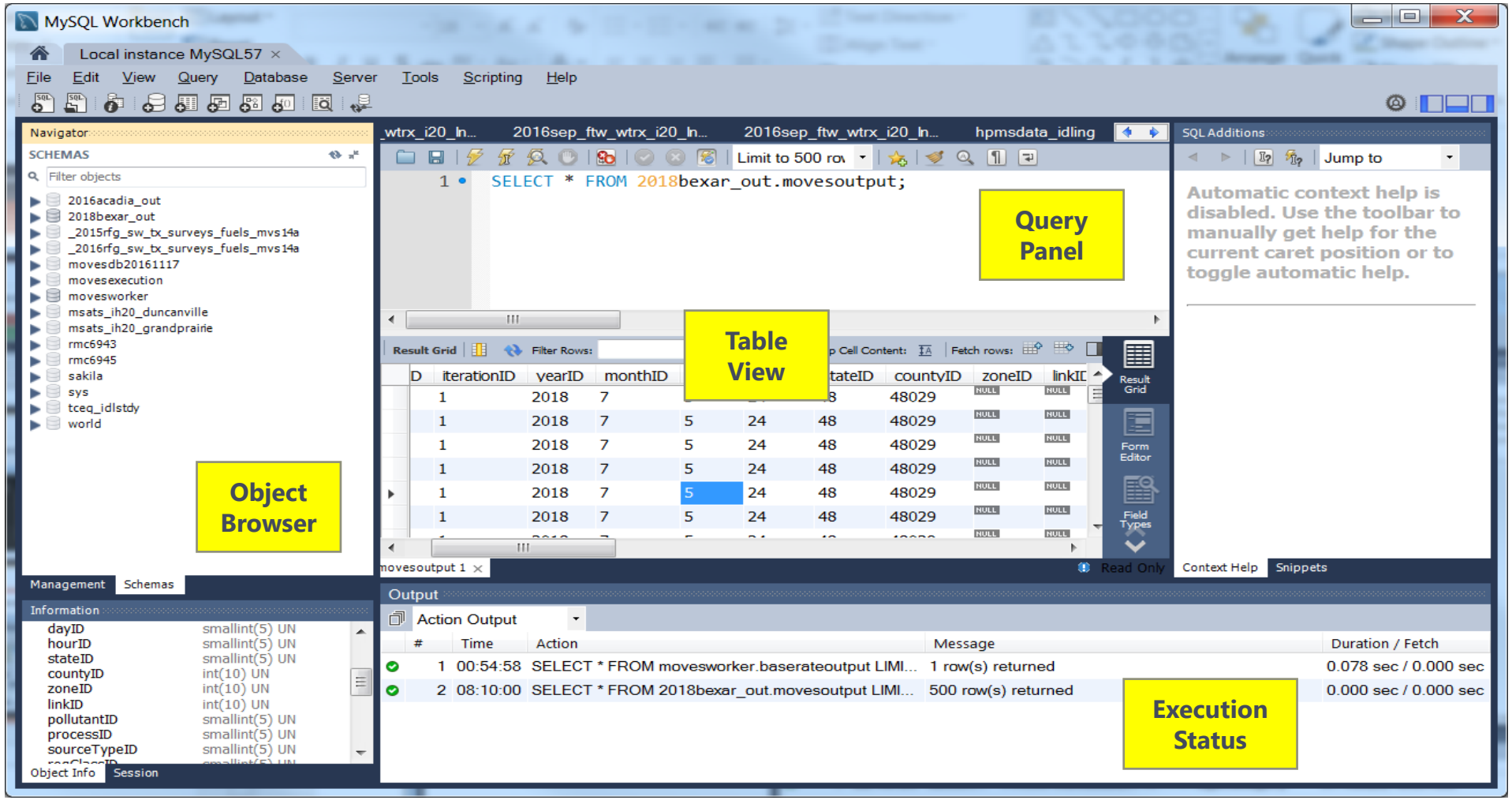
Click *Connect*

- First time in Workbench
  - Need to enter the root user password after clicking *Connect*. This password was set when MySQL was installed
- If you do not know this password
  - right-click on *Local instance MySQL*, select *Edit Configuration…*, set Username to **moves**, and click *Close*. When prompted for a password, enter **moves**

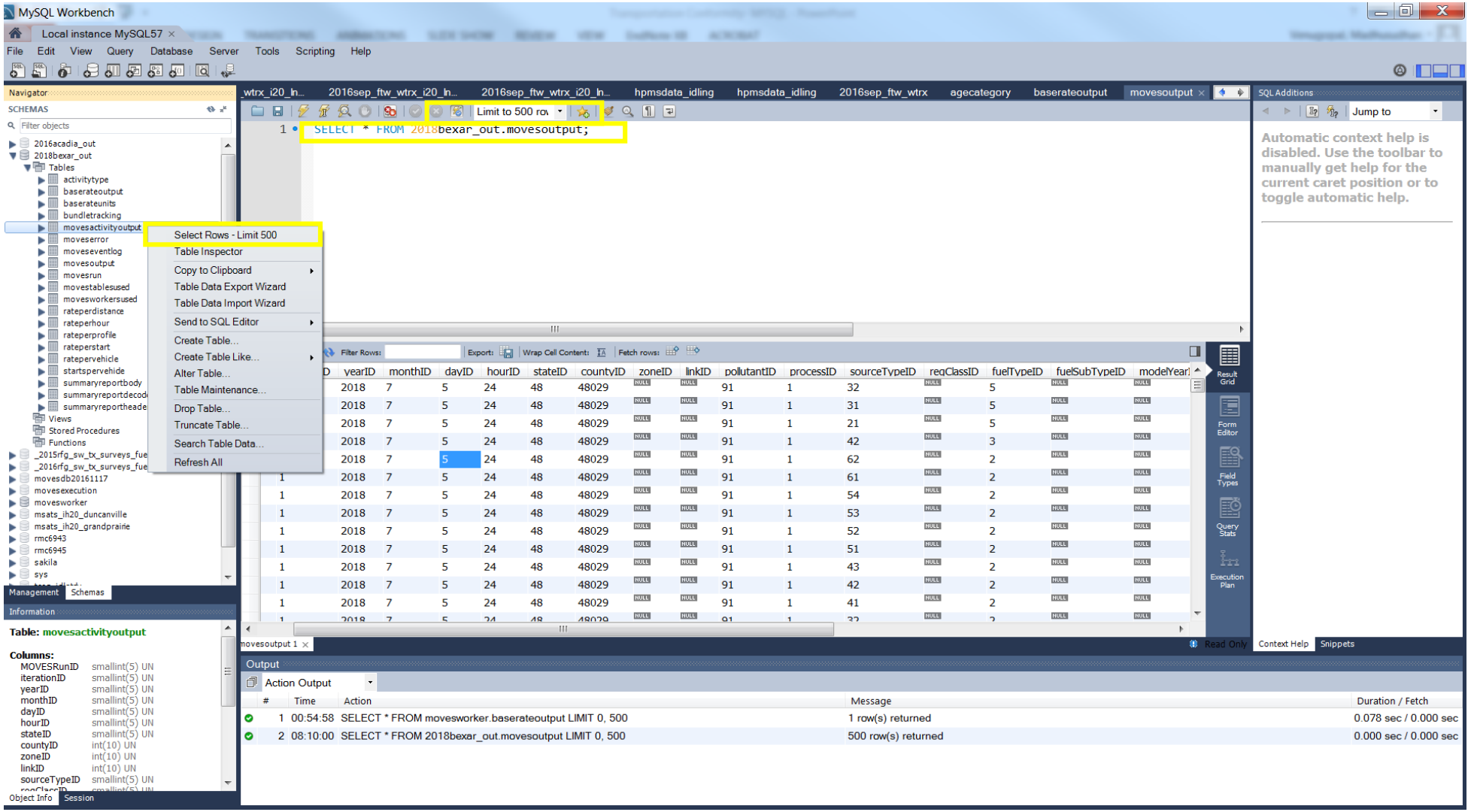
MYSQL Workbench Connection Cont'd



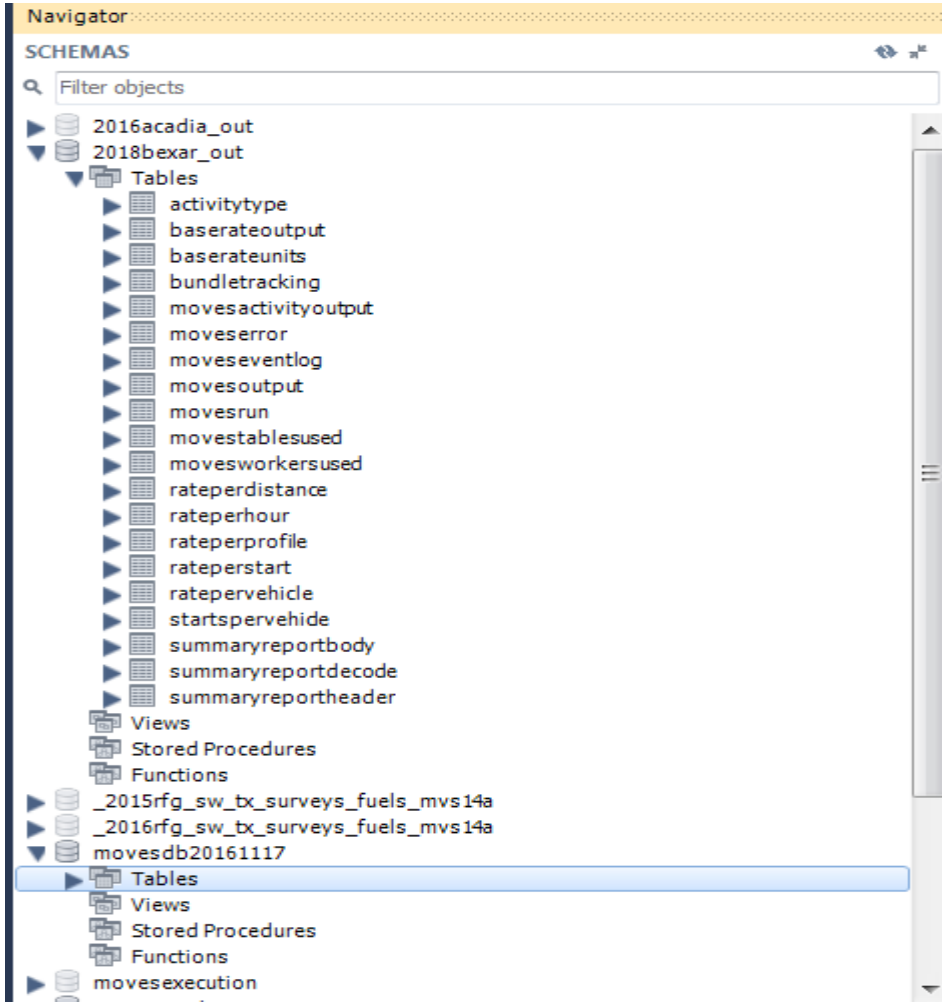
MYSQL Workbench Overview

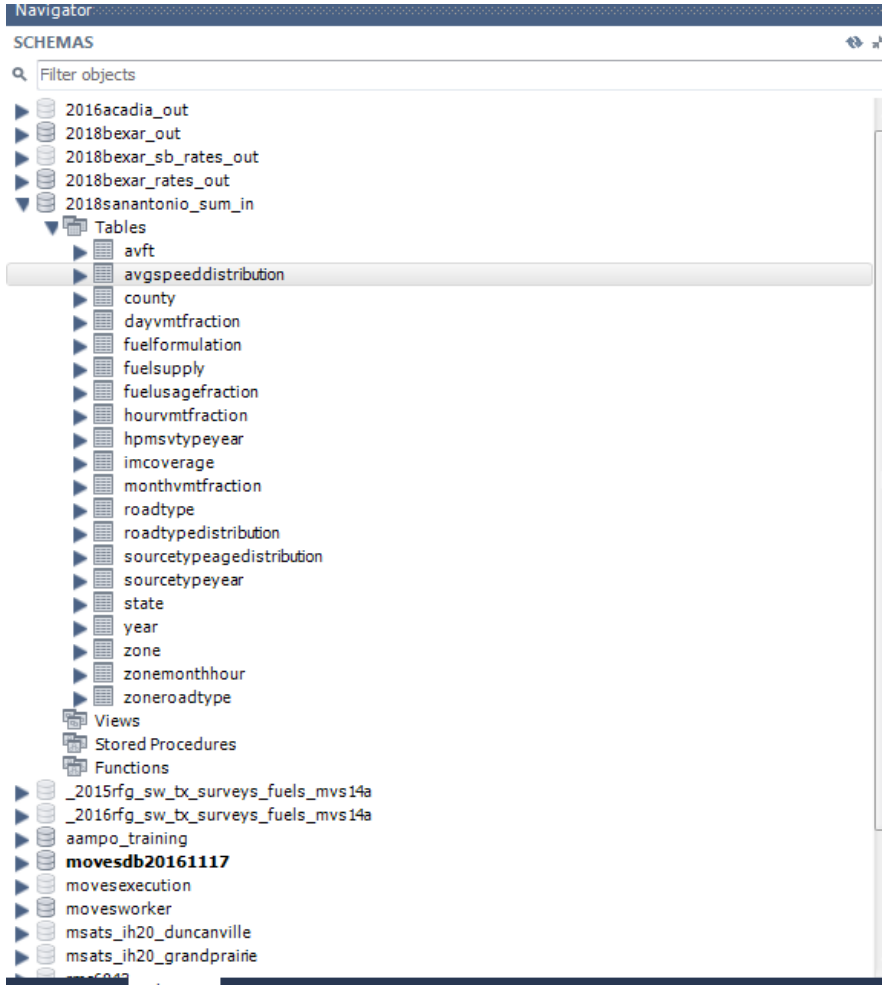


MYSQL Workbench Overview



MYSQL Workbench: Exploring Inputs and Outputs





MYSQL QUERY: Basics

```
SELECT * FROM table_name;
```

MYSQL QUERY

- Used to display or aggregate output database data
- Used to load, alter table, update data
- Syntax must be followed properly to avoid errors.
- '*' Can be replaced with column_name(s) separated by ','
- Multiple commands can be used to complete a query
- Commas can be used to separate multiple fields
- Table to be queried -the syntax is database name followed by "." and the table name

Basic MYSQL QUERY: SELECT

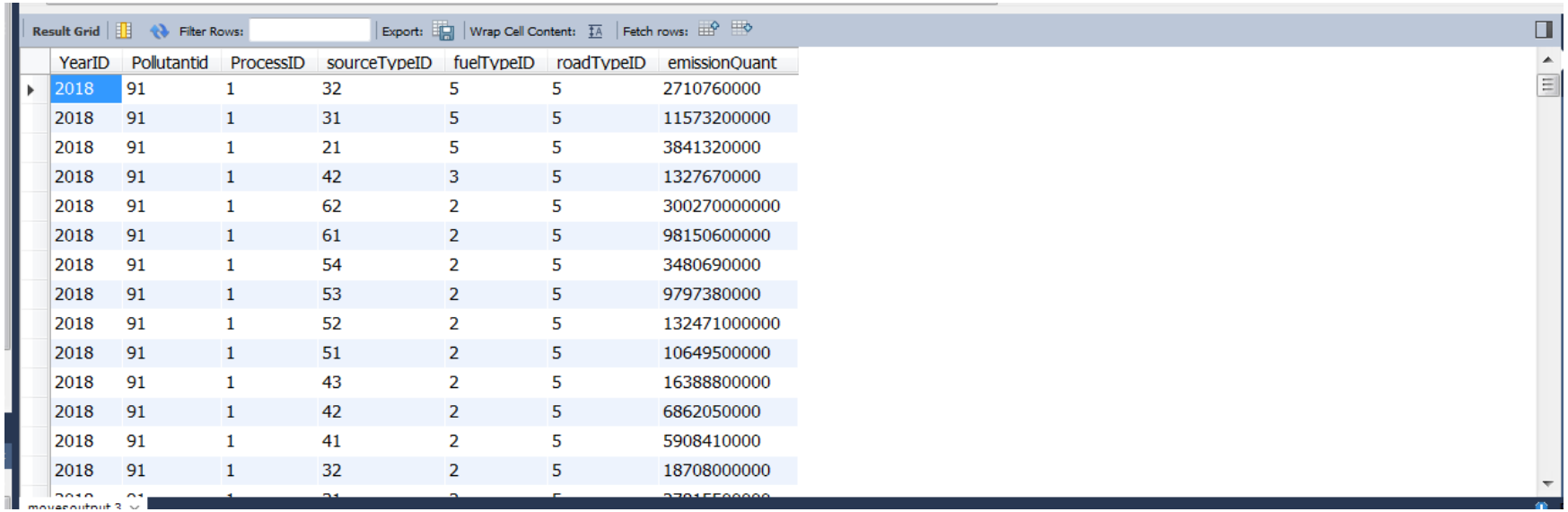
```
SELECT * FROM 2018bexar_out.movesoutput;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

MOVESRunID	iterationID	yearID	monthID	dayID	hourID	stateID	countyID	zoneID	linkID	pollutantID	processID	sourceTypeID	regClassID	fuelTypeID	fuelSubTypeID
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	32	NULL	5	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	31	NULL	5	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	21	NULL	5	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	42	NULL	3	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	62	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	61	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	54	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	53	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	52	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	51	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	43	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	42	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	41	NULL	2	NULL
1	1	2018	7	5	24	48	48029	NULL	NULL	91	1	32	NULL	2	NULL

Basic MYSQL QUERY: SELECT

```
SELECT YearID, Pollutantid, ProcessID, sourceTypeID,
fuelTypeID,roadTypeID,emissionQuant FROM
2018bexar_out.movesoutput;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

YearID	Pollutantid	ProcessID	sourceTypeID	fuelTypeID	roadTypeID	emissionQuant
2018	91	1	32	5	5	2710760000
2018	91	1	31	5	5	11573200000
2018	91	1	21	5	5	3841320000
2018	91	1	42	3	5	1327670000
2018	91	1	62	2	5	300270000000
2018	91	1	61	2	5	98150600000
2018	91	1	54	2	5	3480690000
2018	91	1	53	2	5	9797380000
2018	91	1	52	2	5	132471000000
2018	91	1	51	2	5	10649500000
2018	91	1	43	2	5	16388800000
2018	91	1	42	2	5	6862050000
2018	91	1	41	2	5	5908410000
2018	91	1	32	2	5	18708000000

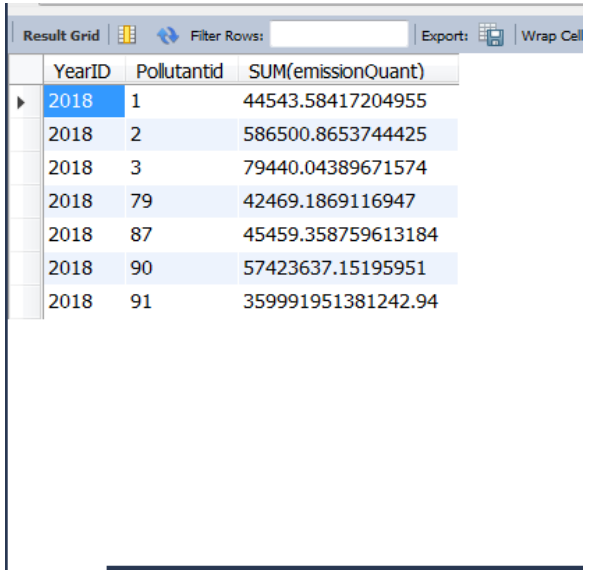
Basic MYSQL QUERY: SELECT With Aggregation

- **COUNT** - Counts the number of observations.
- **SUM** - Calculates the sum of observations.
- **MIN/MAX** - Calculates the min/max
- **AVG** - Calculates the average

Note: Use **Group By** along with these functions

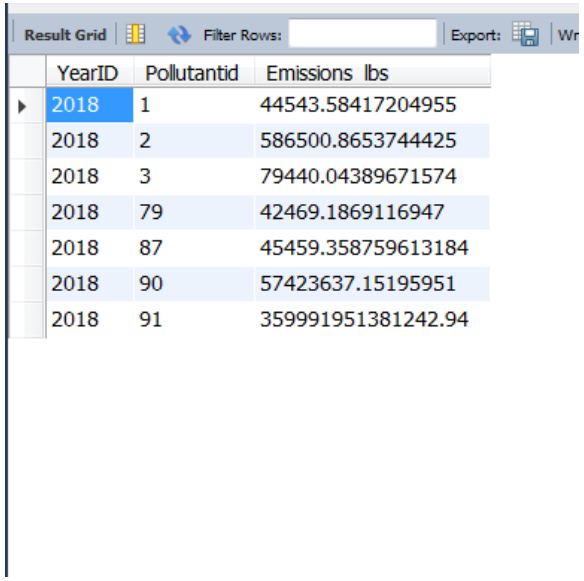
Basic MYSQL QUERY: SELECT With Aggregation

```
SELECT YearID, Pollutantid, SUM(emissionQuant) as
Emissions_lbs FROM 2018bexar_out.movesoutputgroup by
YearID, Pollutantid;
```



Result Grid | Filter Rows: | Export: | Wrap Cell

YearID	Pollutantid	SUM(emissionQuant)
2018	1	44543.58417204955
2018	2	586500.8653744425
2018	3	79440.04389671574
2018	79	42469.1869116947
2018	87	45459.358759613184
2018	90	57423637.15195951
2018	91	359991951381242.94



Result Grid | Filter Rows: | Export: | Wr

YearID	Pollutantid	Emissions_lbs
2018	1	44543.58417204955
2018	2	586500.8653744425
2018	3	79440.04389671574
2018	79	42469.1869116947
2018	87	45459.358759613184
2018	90	57423637.15195951
2018	91	359991951381242.94

Basic MYSQL QUERY: SELECT With Order By

Used to order a variable in descending or ascending order

```
SELECT YearID, Pollutantid, SUM(emissionQuant) as
Emissions_lbs FROM 2018bexar_out.movesoutputgroup by
YearID, Pollutantid order by pollutantid desc;
```

Result Grid | Filter Rows: | Export: | Wrap

YearID	Pollutantid	Emissions_lbs
2018	91	359991951381242.94
2018	90	57423637.15195951
2018	87	45459.358759613184
2018	79	42469.1869116947
2018	3	79440.04389671574
2018	2	586500.8653744425
2018	1	44543.58417204955

Basic MYSQL QUERY: Comparison Operators

Usually Used with WHERE

Includes operators

= , != , < , > , <= , >=

```
SELECT Yearid, Monthid, PollutantID, processid,
sourcetypeid, fueltypeid, roadtypeid, emissionQuant
FROM 2018bexar_out.movesoutputwhere processID = 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Yearid	Monthid	PollutantID	processid	sourcetypeid	fueltypeid	roadtypeid	emissionQuant
2018	7	91	1	32	5	5	2710760000
2018	7	91	1	31	5	5	11573200000
2018	7	91	1	21	5	5	3841320000
2018	7	91	1	42	3	5	1327670000
2018	7	91	1	62	2	5	300270000000
2018	7	91	1	61	2	5	98150600000
2018	7	91	1	54	2	5	3480690000
2018	7	91	1	53	2	5	9797380000
2018	7	91	1	52	2	5	132471000000
2018	7	91	1	51	2	5	10649500000
2018	7	91	1	43	2	5	16388800000
2018	7	91	1	42	2	5	6862050000
2018	7	91	1	41	2	5	5908410000

Basic MYSQL QUERY: Comparison Operators

Normally Used with WHERE

Includes operators

= , != , < , > , <= , >=

```
SELECT Yearid, Monthid, PollutantID, processid,
sourcetypeid, fueltypeid, roadtypeid, emissionQuant
FROM 2018bexar_out.movesoutputwhere processID = 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Yearid	Monthid	PollutantID	processid	sourcetypeid	fueltypeid	roadtypeid	emissionQuant
2018	7	91	1	32	5	5	2710760000
2018	7	91	1	31	5	5	11573200000
2018	7	91	1	21	5	5	3841320000
2018	7	91	1	42	3	5	1327670000
2018	7	91	1	62	2	5	300270000000
2018	7	91	1	61	2	5	98150600000
2018	7	91	1	54	2	5	3480690000
2018	7	91	1	53	2	5	9797380000
2018	7	91	1	52	2	5	132471000000
2018	7	91	1	51	2	5	10649500000
2018	7	91	1	43	2	5	16388800000
2018	7	91	1	42	2	5	6862050000
2018	7	91	1	41	2	5	5908410000

Basic MYSQL QUERY: Comparison Operators

SELECT Yearid, Monthid, PollutantID, processid, sourcetypeid, fueltypeid, roadtypeid, emissionQuant FROM 2018bexar_out.movesoutput where processID > 10;

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Yearid	Monthid	PollutantID	processid	sourcetypeid	fueltypeid	roadtypeid	emissionQuant
2018	7	91	91	62	2	1	3247940000
2018	7	91	90	62	2	1	77158200000
2018	7	90	91	62	2	1	527.45
2018	7	90	90	62	2	1	12530.1
2018	7	87	19	32	5	5	0.0164225
2018	7	87	19	31	5	5	0.0699305
2018	7	87	19	21	5	5	0.0231446
2018	7	87	19	62	2	5	1.50479
2018	7	87	19	61	2	5	0.473886
2018	7	87	19	54	2	5	0.0174433
2018	7	87	19	53	2	5	0.0387979
2018	7	87	19	52	2	5	0.474517
2018	7	87	19	51	2	5	0.0519401

Basic MYSQL QUERY: Logical Operators

- Traditional Operators

  AND, OR, NOT

- Other Operators
  - **LIKE** - Used to extract similar values and not exact values
  - **IN** - Used to specify the list of values to extract or leave out from a variable
  - **BETWEEN** - Provide condition based on variable in the table
  - **IS NULL** - Allows you to extract data with missing values

Basic MYSQL QUERY: Logical Operators

SELECT Yearid, Monthid, PollutantID, processid, sourcetypeid, fueltypeid, roadtypeid, emissionQuant FROM 2018bexar_out.movesoutputwhere processID > 10 and pollutantid = 3;

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Yearid	Monthid	PollutantID	processid	sourcetypeid	fueltypeid	roadtypeid	emissionQuan
2018	7	3	15	32	5	5	0.000003132(
2018	7	3	15	31	5	5	0.000013772
2018	7	3	15	21	5	5	0.000003940
2018	7	3	15	42	3	5	0.000017855
2018	7	3	15	62	2	5	0.093585
2018	7	3	15	61	2	5	0.0254784
2018	7	3	15	54	2	5	0.00127266
2018	7	3	15	53	2	5	0.00176554
2018	7	3	15	52	2	5	0.0285823
2018	7	3	15	51	2	5	0.00268263
2018	7	3	15	43	2	5	0.0098479
2018	7	3	15	42	2	5	0.00424895
2018	7	3	15	41	2	5	0.00381257

SELECT Yearid, hourid, PollutantID, processid, sourcetypeid, fueltypeid, roadtypeid, emissionQuant FROM 2018bexar_out.movesoutputwhere sourceTypeID = 21 and hourid = 1 and (processID = 1 or processID = 2);

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Yearid	hourid	PollutantID	processid	sourcetypeid	fueltypeid	roadtypeid	emissionQuant
2018	1	91	1	21	5	5	2057780000
2018	1	91	1	21	2	5	4778520000
2018	1	91	1	21	1	5	561641000000
2018	1	91	1	21	5	4	1526120000
2018	1	91	1	21	2	4	3555020000
2018	1	91	1	21	1	4	416293000000
2018	1	91	1	21	5	3	90002100
2018	1	91	1	21	2	3	209529000
2018	1	91	1	21	1	3	24556600000
2018	1	91	1	21	5	2	118746000
2018	1	91	1	21	2	2	276570000
2018	1	91	1	21	1	2	32388500000
2018	1	91	2	21	5	1	43207900

movesoutput 22

Basic MYSQL QUERY: Logical Operator BETWEEN

```
SELECT Yearid, hourid, PollutantID, processid,
sourcetypeid, fueltypeid, roadtypeid, emissionQuant FROM
2018bexar_out.movesoutputwhere sourceTypeID between
20 and 40 and roadtypeid = 5;
```

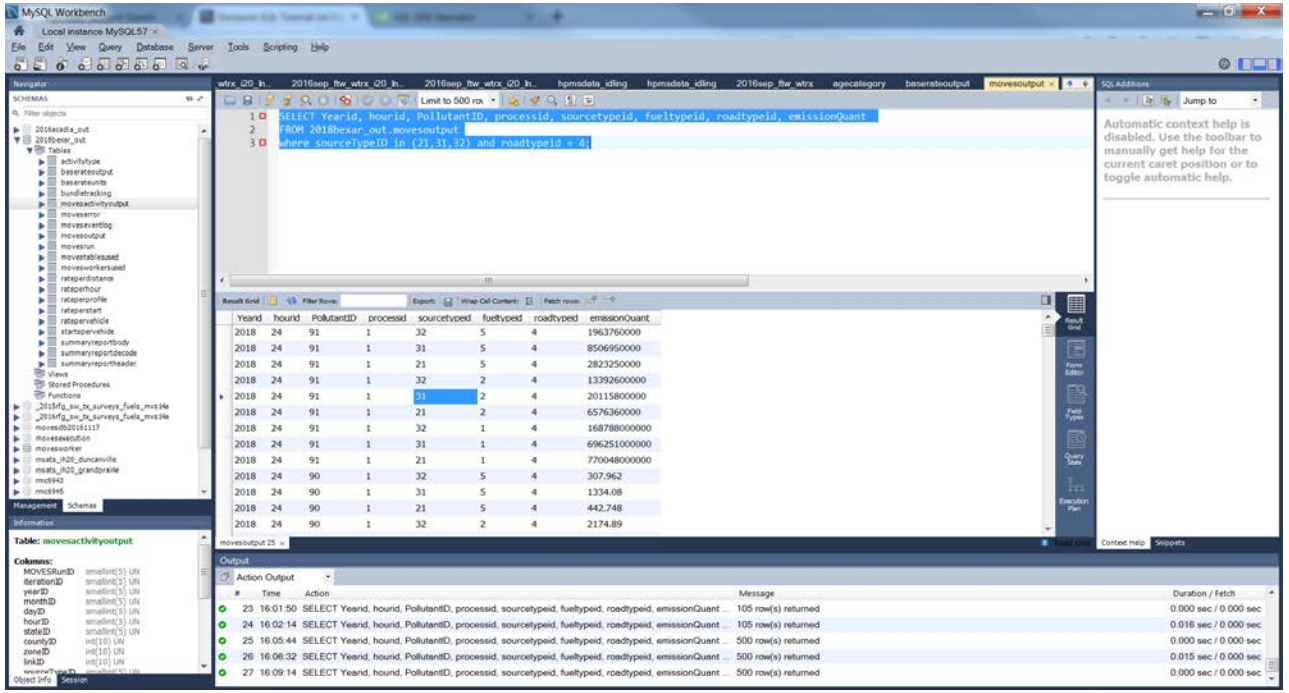
Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Yearid	hourid	PollutantID	processid	sourcetypeid	fueltypeid	roadtypeid	emissionQuant
2018	24	91	1	32	5	5	2710760000
2018	24	91	1	31	5	5	11573200000
2018	24	91	1	21	5	5	3841320000
2018	24	91	1	32	2	5	18708000000
2018	24	91	1	31	2	5	27815500000
2018	24	91	1	21	2	5	8918690000
2018	24	91	1	32	1	5	233084000000
2018	24	91	1	31	1	5	947721000000
2018	24	91	1	21	1	5	1048300000000
2018	24	90	1	32	5	5	425.109
2018	24	90	1	31	5	5	1814.93
2018	24	90	1	21	5	5	602.405
2018	24	90	1	32	2	5	3038.1

movesoutput 24

Basic MYSQL QUERY: Logical Operator IN

```
SELECT Yearid, hourid, PollutantID, processid, sourcetypeid, fueltypeid,
roadtypeid, emissionQuant FROM 2018bexar_out.movesoutputwhere
sourceTypeID in (21,31,32) and roadtypeid = 4;
```



Basic MYSQL QUERY: Logical Operator LIKE

```
SELECT Yearid, sourcetypeid, sum(emissionQuant)/2000 as
VOC_tons FROM 2018bexar_out.movesoutput where
sourcetypeid like '_1' and pollutantID = 79
group by Yearid, sourcetypeid;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

Yearid	sourcetypeid	VOC_tons
2018	11	1.035070551612218
2018	21	8.103811439216615
2018	31	7.7462094310709055
2018	41	0.016260798665449328
2018	51	0.023939553606991466
2018	61	0.17725860813658734

```
SELECT Yearid, sourcetypeid, sum(emissionQuant)/2000
as VOC_tons FROM 2018bexar_out.movesoutputwhere
sourcetypeid like '%1' and pollutantID = 79
group by Yearid, sourcetypeid;
```

Note: Function that can be extensively used to query strings or characters

Exercise 1

Develop a query with following fields and format

(a)

Result Grid | Filter Rows: | Export:

Yearid	sourcetypeid	NOx tons
2018	62	13.35
2018	61	2.99
2018	54	0.18
2018	53	0.20
2018	52	3.42
2018	51	0.34
2018	43	0.77
2018	42	0.36
2018	41	0.26
2018	32	2.26
2018	31	8.64
2018	21	6.67
2018	11	0.27

Result 3

(b)

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Yearid	sourcetypeid	fueltypeid	roadtypeid	NOx tons
2018	21	1	1	2.45
2018	21	2	1	0.01
2018	21	2	2	0.00
2018	21	1	2	0.14
2018	21	2	3	0.00
2018	21	1	3	0.09
2018	21	2	4	0.01
2018	21	1	4	1.91
2018	21	2	5	0.01
2018	21	1	5	2.03

Exercise 1 – Answer Key

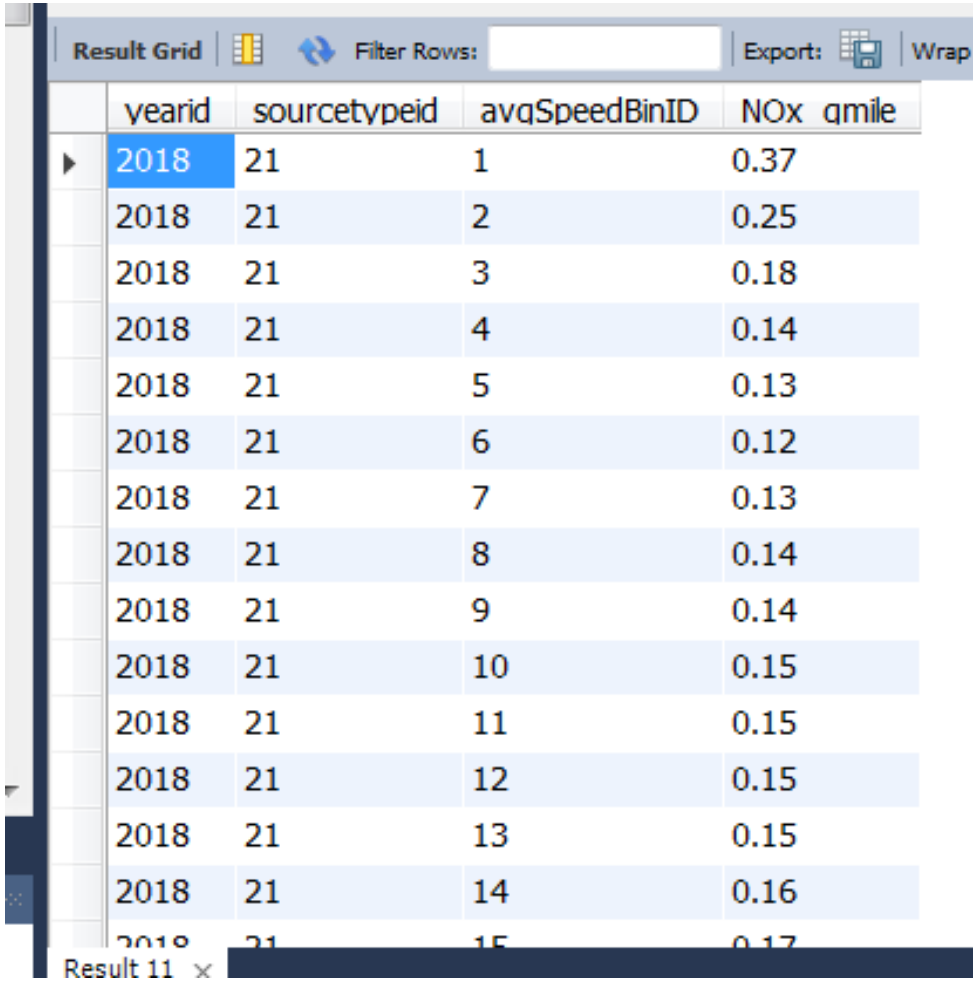
```
SELECT Yearid, sourcetypeid, SUM(emissionQuant)/2000 as
NOx_tons FROM 2018bexar_out.movesoutputwhere Pollutantid = 3
group by Yearid, sourcetypeid order by sourceTypeID desc;
```

```
SELECT Yearid, sourcetypeid, fueltypeid, roadtypeid,
ROUND(SUM(emissionQuant)/2000,2) as NOx_tonsFROM
2018bexar_out.movesoutputwhere Pollutantid = 3 and sourcetypeid =
21 and fueltypeid in (1,2) group by Yearid, fueltypeid, roadTypeIDorder
by roadTypeID asc;
```

Exercise - 2

Request has been made for

- Light Duty Gasoline Vehicles
- AM peak (8 AM)
- Rate per Distance (includes all processes)
- NOx



yearid	sourcetypeid	avgSpeedBinID	NOx gmile
2018	21	1	0.37
2018	21	2	0.25
2018	21	3	0.18
2018	21	4	0.14
2018	21	5	0.13
2018	21	6	0.12
2018	21	7	0.13
2018	21	8	0.14
2018	21	9	0.14
2018	21	10	0.15
2018	21	11	0.15
2018	21	12	0.15
2018	21	13	0.15
2018	21	14	0.16

Exercise 2 – Answer Key

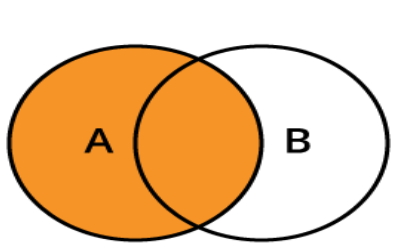
```
SELECT yearid, sourcetypeid, avgSpeedBinID,
Round(sum(rateperdistance),2) as running_rate_gmile    FROM
2018bexar_rates_out.rateperdistancewhere sourcetypeid = 21 and
hourid = 9 and roadtypeid = 4 and fueltypeid = 1 and pollutantid
= 3 group by yearid, sourcetypeid, avgSpeedBinID;
```

Exercise 3

Request has been made to identify what Bexar County light duty gasoline vehicle starts by hour of day and associated start VOC emissions

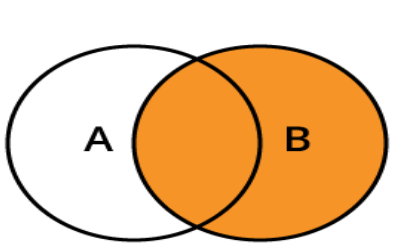
Need to run two queries on different tables

Advanced MYSQL QUERY: JOINS

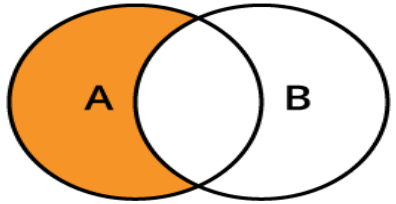


SELECT <select_list>
FROM TableA
LEFT JOIN TableB
ON A.Key = B.Key

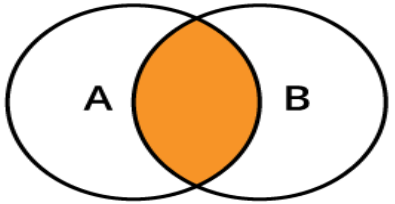
SQL JOIN QUERIES



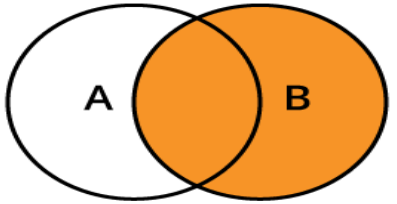
SELECT <select_list>
FROM TableA
RIGHT JOIN TableB
ON A.Key = B.Key



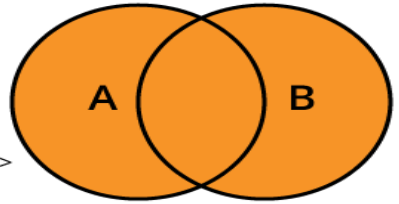
SELECT <select_list>
FROM TableA
LEFT JOIN TableB
ON A.Key = B.Key
WHERE B.Key IS NULL



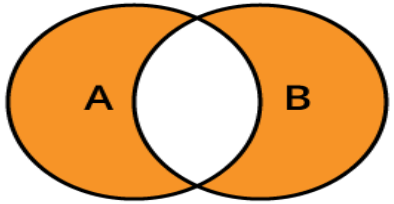
SELECT <select_list>
FROM TableA
INNER JOIN TableB
ON A.Key = B.Key



SELECT <select_list>
FROM TableA
RIGHT JOIN TableB
ON A.Key = B.Key
WHERE A.Key IS NULL



SELECT <select_list>
FROM TableA
FULL OUTER JOIN TableB
ON A.Key = B.Key



SELECT <select_list>
FROM TableA
FULL OUTER JOIN TableB
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS Null

hackerearth

Advanced MYSQL QUERY: JOINS Cont'd

- **INNER JOIN** - Returns the common rows specifying the joining criteria from A and B.
- **OUTER JOIN** - Returns the rows which are not common to A and B.
- **LEFT JOIN** - Returns the rows which are in A but not in B.
- **RIGHT JOIN** - Returns the rows which are in B but not in A**.**
- **FULL OUTER JOIN** - Returns all the rows from both tables.

Advanced MYSQL QUERY: LEFT JOIN

```
SELECT A.Yearid, A.SourceTypeID, B.sourceTypeName,
ROUND(sum(emissionQuant)/2000,2) as VOC_tons
FROM 2018bexar_out.movesoutput A LEFT JOIN
movesdb20161117.sourceusetype B on A.SourceTypeID =
B.sourceTypeIDwhere A.pollutantID = 79 group by A.Yearid,
A.SourceTypeID, B.sourceTypeName
Order by A.SourceTypeID ASC ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Yearid	SourceTypeID	sourceTypeName	VOC_tons
2018	11	Motorcycle	1.04
2018	21	Passenger Car	8.10
2018	31	Passenger Truck	7.75
2018	32	Light Commercial Truck	1.79
2018	41	Intercity Bus	0.02
2018	42	Transit Bus	0.03
2018	43	School Bus	0.11
2018	51	Refuse Truck	0.02
2018	52	Single Unit Short-haul Truck	1.01
2018	53	Single Unit Long-haul Truck	0.04
2018	54	Motor Home	0.12
2018	61	Combination Short-haul Tr...	0.18
2018	62	Combination Long-haul Tru...	1.03

Exercise - 4

Using Left Join example from previous slide add fuel type and fuel type description to the table

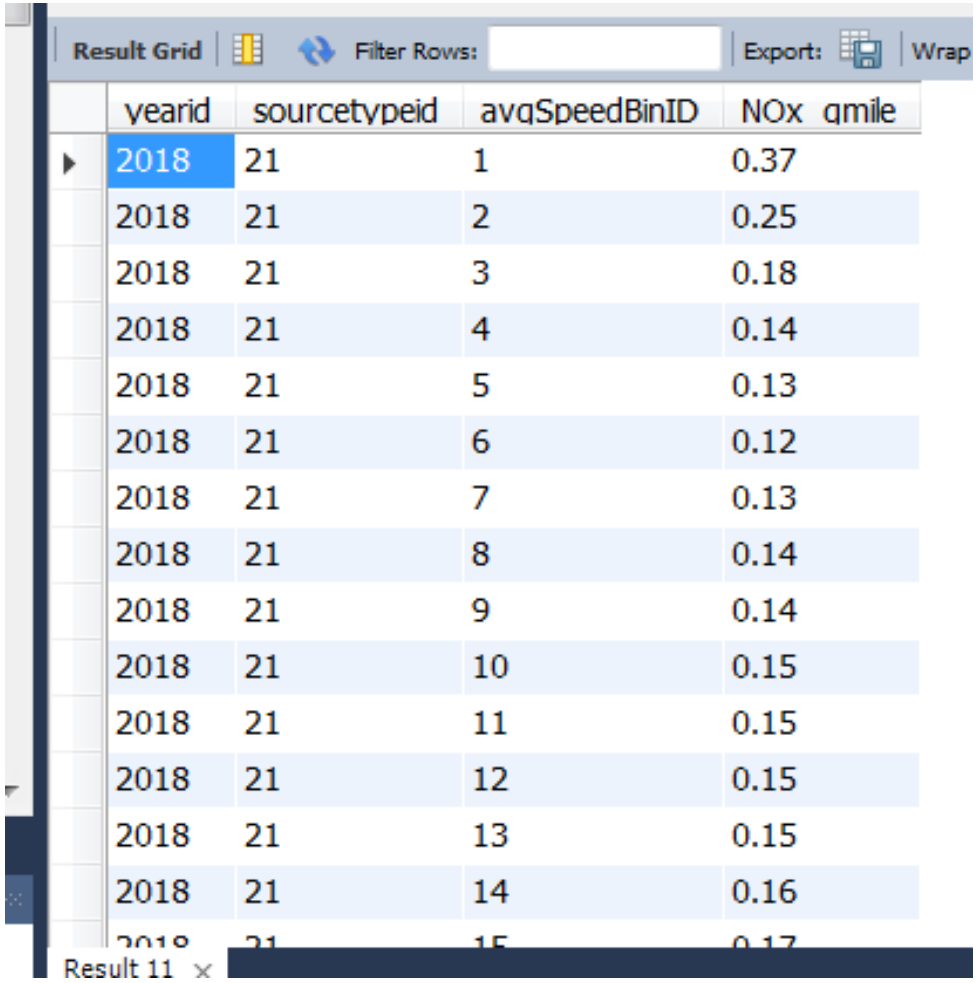
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Yearid	SourceTypeID	fuelTypeID	sourceTypeName	fuelTypeDesc	VOC_tons
2018	11	1	Motorcycle	Gasoline	1.04
2018	21	1	Passenger Car	Gasoline	8.07
2018	21	2	Passenger Car	Diesel Fuel	0.01
2018	21	5	Passenger Car	Ethanol (E-85)	0.02
2018	31	1	Passenger Truck	Gasoline	7.64
2018	31	2	Passenger Truck	Diesel Fuel	0.07
2018	31	5	Passenger Truck	Ethanol (E-85)	0.04
2018	32	1	Light Commercial Truck	Gasoline	1.72
2018	32	2	Light Commercial Truck	Diesel Fuel	0.06
2018	32	5	Light Commercial Truck	Ethanol (E-85)	0.01
2018	41	2	Intercity Bus	Diesel Fuel	0.02
2018	42	1	Transit Bus	Gasoline	0.00
2018	42	2	Transit Bus	Diesel Fuel	0.02
2018	42	3	Transit Bus	Compressed ...	0.00
2018	43	1	School Bus	Gasoline	0.02
2018	43	2	School Bus	Diesel Fuel	0.09
2018	51	1	Refuse Truck	Gasoline	0.00
2018	51	2	Refuse Truck	Diesel Fuel	0.02
2018	52	1	Single Unit Short-haul Truck	Gasoline	0.69
2018	52	2	Single Unit Short-haul Truck	Diesel Fuel	0.32

Exercise 5

Using left Join example, add fuel type and fuel type description to the table shown on the right

Note: Your Table Yearid will be 2010

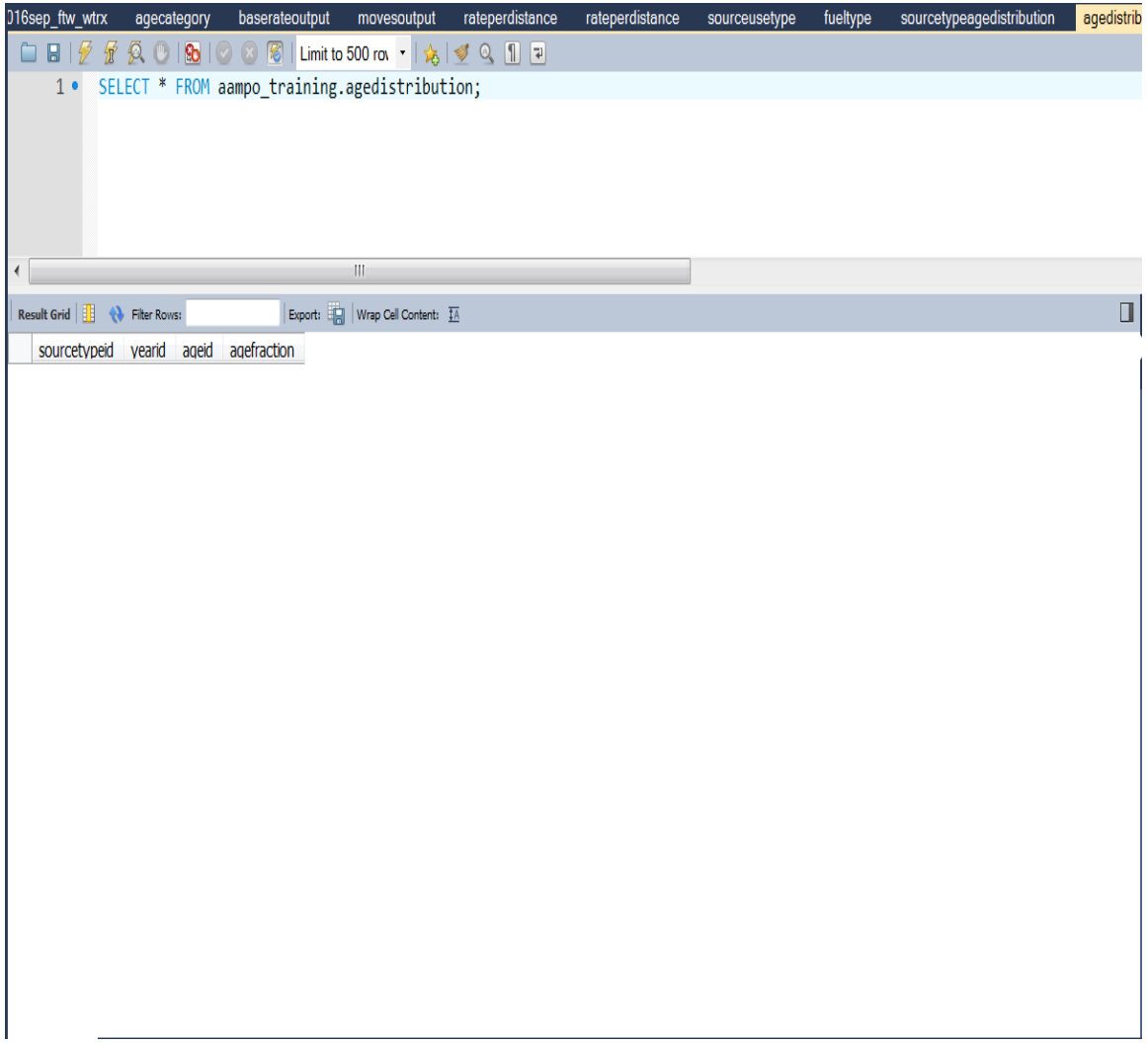


yearid	sourcetypeid	avgSpeedBinID	NOx gmile
2018	21	1	0.37
2018	21	2	0.25
2018	21	3	0.18
2018	21	4	0.14
2018	21	5	0.13
2018	21	6	0.12
2018	21	7	0.13
2018	21	8	0.14
2018	21	9	0.14
2018	21	10	0.15
2018	21	11	0.15
2018	21	12	0.15
2018	21	13	0.15
2018	21	14	0.16

Advanced MYSQL QUERY: Create DB & Table

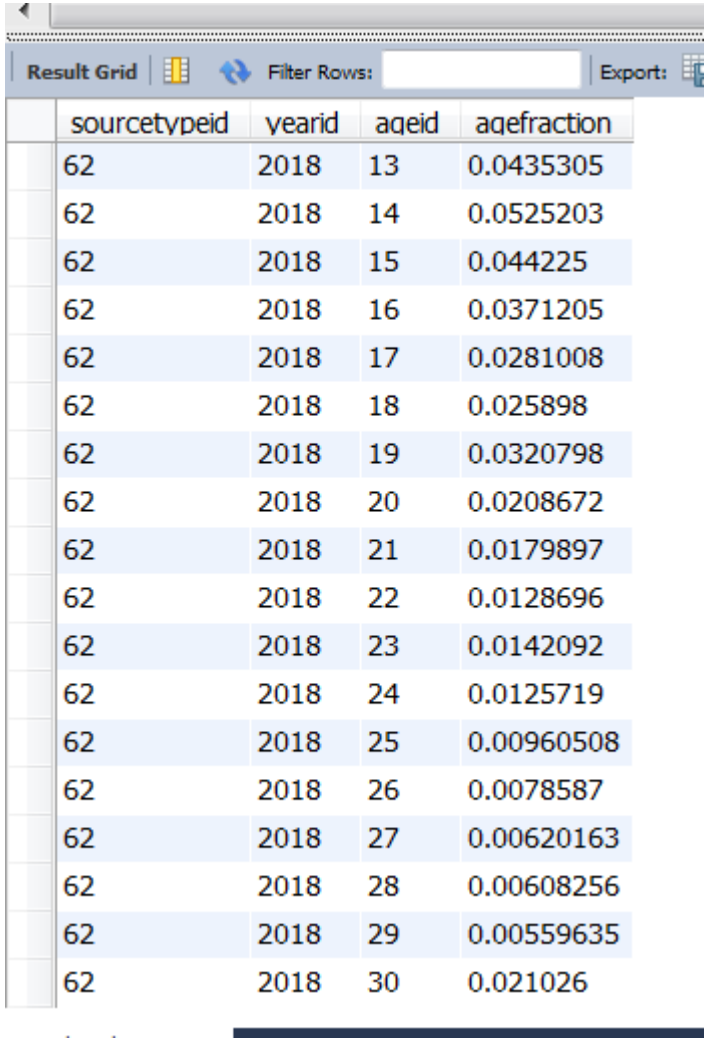
CREATE DATABASE AAMPO_Training;

CREATE table AAMPO_Training.agedistribution(sourcetypeid int,yearid int,ageid int,agefraction float);



Advanced MYSQL QUERY: Load data

```
LOAD DATA LOCAL INFILE
'C:/Presentations/San Antonio MOVES
Training/Module-4/Inputs/2018Bexar_AgeDist.csv'
INTO TABLE aampo_training.agedistribution
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
Ignore 1 lines ;
```



Result Grid | Filter Rows: | Export:

sourcetypeid	yearid	ageid	agefraction
62	2018	13	0.0435305
62	2018	14	0.0525203
62	2018	15	0.044225
62	2018	16	0.0371205
62	2018	17	0.0281008
62	2018	18	0.025898
62	2018	19	0.0320798
62	2018	20	0.0208672
62	2018	21	0.0179897
62	2018	22	0.0128696
62	2018	23	0.0142092
62	2018	24	0.0125719
62	2018	25	0.00960508
62	2018	26	0.0078587
62	2018	27	0.00620163
62	2018	28	0.00608256
62	2018	29	0.00559635
62	2018	30	0.021026

Exercise 6

Create table averagespeeddistribution

Load data from csv files in Module-4 input folder

Advanced MYSQL QUERY: Create and Load data

CREATE table AAMPO_Training.agedistribution **like** movesdb20161117.sourcetypeagedistribution;

```
LOAD DATA LOCAL INFILE
'C:/Presentations/San Antonio MOVES
Training/Module-4/Inputs/2018Bexar_AgeDist.csv'
INTO TABLE aampo_training.agedistribution
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
Ignore 1 lines ;
```

Exercise 7

Create table hourVMTfraction using **Like** function

Load data from csv files in Module-4 input folder

Advanced MYSQL QUERY: ALTER and UPDATE

Alter Table used to add, delete, or modify columns in an existing table

Alter table aampo_training.agedistribution
ADD column veh_pop int,
ADD column FleetAge float;

Update table SET with formulas

Key Pointers

- Data aggregation queries most used
- Attention needed to modification queries such as alter, update, etc.
- Make another copy of MOVES default database
- Save all your queries in a text file with description

Resources

- MOVES Training Sessions: https://www.epa.gov/moves/moves-training-sessions
- FHWA Training: https://www.fhwa.dot.gov/resourcecenter/teams/planningair/mysqlwebinar.cfm
- Web Sources:
  - www.w3schools.com
  - http://www.sqlcourse.com/table.html
  - Others